Apply RNNs to text and sequence data

Assignment-4

Rajesh Nalliboyina

Department of Information Systems and Business Analytics,

Kent State University

BA-64061-001 Advanced Machine Learning

Chaojiang (CJ) Wu, Ph.D.

11-26-2024

1. Purpose

The objective of this assignment is to explore the application of Recurrent Neural Networks (RNNs) for analyzing text and sequence data. The assignment aims to:

1. Apply RNNs to IMDB sentiment analysis data.
2. Demonstrate strategies to improve model performance, especially when handling limited data.
3. Evaluate and compare the effectiveness of two approaches:
    - Using a custom embedding layer.
    - Using pretrained word embeddings (GloVe).

2. Dataset Overview

The IMDB dataset, containing labeled movie reviews, was used for this analysis. Key preprocessing steps included:

- **Cutting off reviews after 150 words.**
- **Restricting training samples to 100.**
- **Validating on 10,000 samples.**
- **Considering only the top 10,000 words** in the vocabulary.

3. Model Architectures

Two models were trained for this task:

1.Custom Embedding Model:

Embedding layer with 128 dimensions.

A bidirectional LSTM layer with 32 units.

Dropout for regularization (rate = 0.5).

Dense output layer with a sigmoid activation function.

Layer	Output Shape	Param #
Input	(None, None)	0
Embedding	(None, None, 128)	1,280,000
Bidirectional LSTM	(None, 64)	41,216
Dropout	(None, 64)	0
Dense (Sigmoid)	(None, 1)	65
Total Parameters		**1,321,281**

Pretrained Embedding Model (GloVe):

Pretrained GloVe embeddings (100 dimensions).

Embedding layer initialized with pretrained weights (non-trainable).

A bidirectional LSTM layer with 32 units.

Dropout for regularization (rate = 0.5).

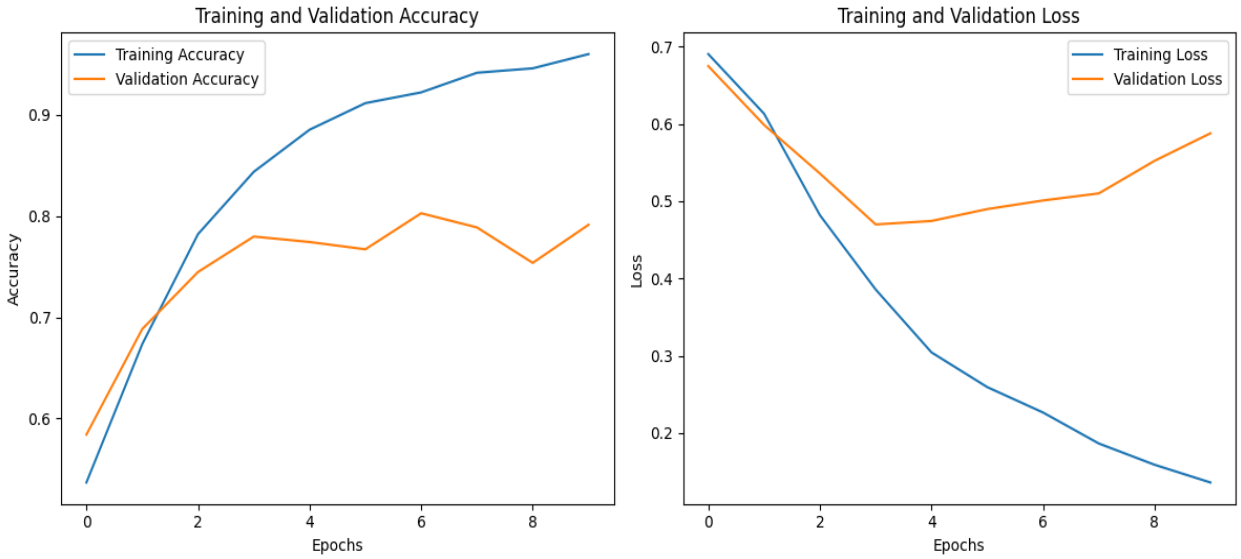
Dense output layer with a sigmoid activation function.

Layer	Output Shape	Param #
Input	(None, None)	0
Embedding (GloVe)	(None, None, 100)	1,000,000
Bidirectional LSTM	(None, 64)	34,048
Dropout	(None, 64)	0
Dense (Sigmoid)	(None, 1)	65
Total Parameters		**1,034,113**

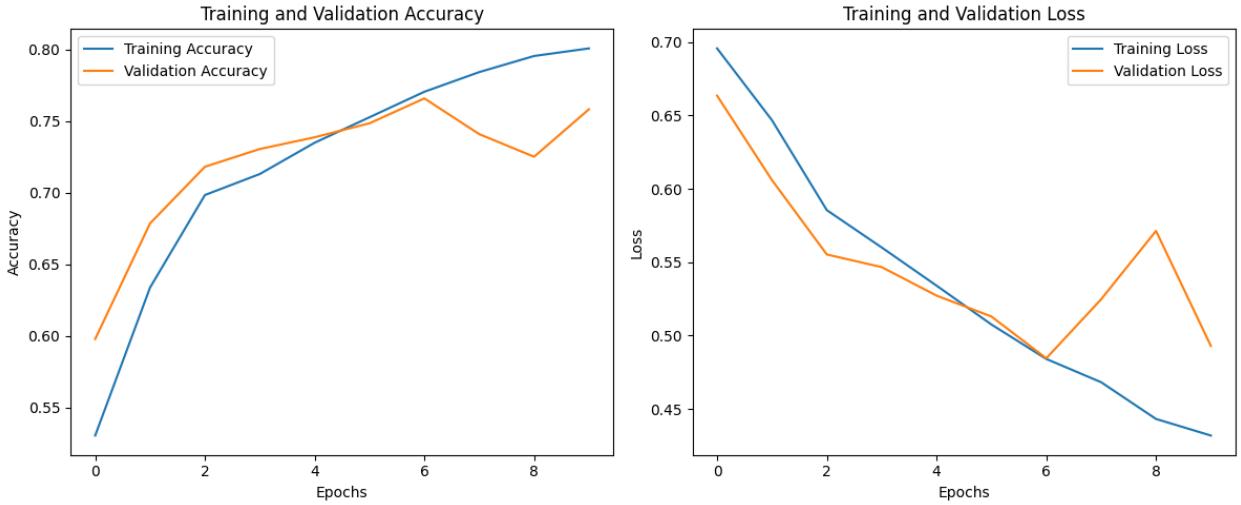
4. Results and Evaluation

a. Training and Validation Accuracy and Loss Plots for both models are presented below, showing trends in training and validation accuracy and loss over 10 epochs.

Custom Embedding Model Accuracy and Loss

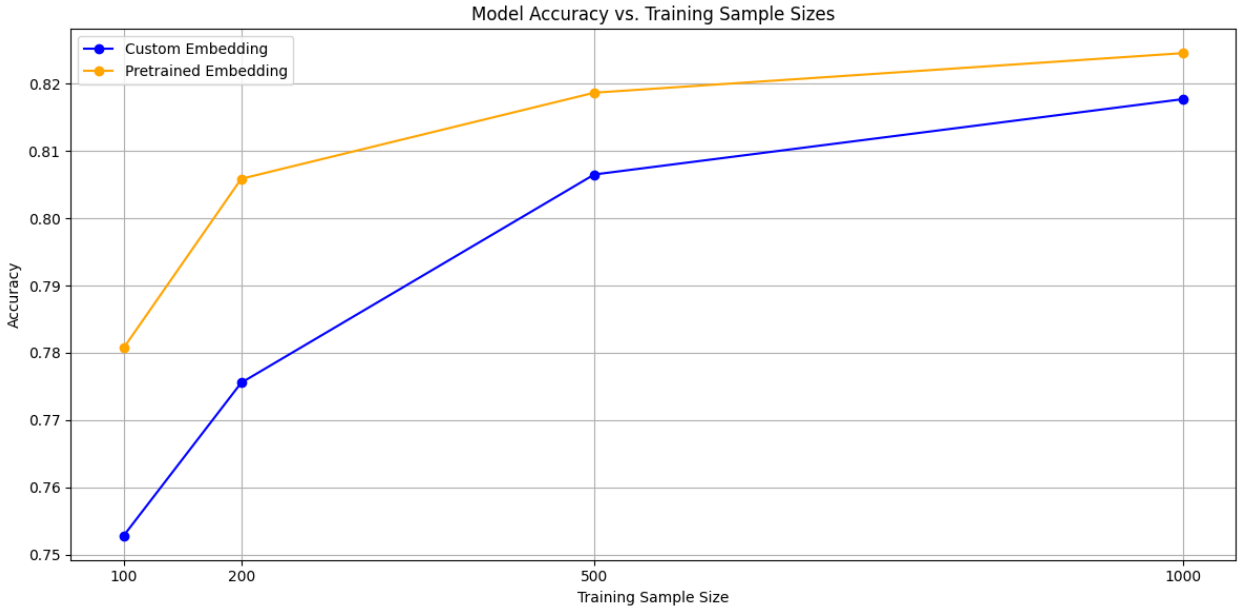


Pretrained Embedding Model Accuracy and Loss



Effect of Sample Size on Model Accuracy To determine how sample size impacts performance, both models were trained on varying sample sizes (100, 200, 500, and 1,000).

Comparison Plot: Final Test Accuracy vs. Sample Size



Sample Size	Custom Embedding Accuracy	Pretrained Embedding Accuracy	Difference
100	0.7528	0.7807	0.0279
200	0.7756	0.8059	0.0303
500	0.8065	0.8187	0.0122
1,000	0.8177	0.8246	0.0069

Analysis of Results

The summary table of accuracies for custom embedding and pretrained embedding models at different training sample sizes highlights distinct trends and provides insights into the performance of the two approaches under varying data constraints.

2. Observations

1.Accuracy Gap Between Custom and Pretrained Models:

Small Sample Sizes (100–200):

The pretrained embeddings significantly outperformed the custom embeddings with differences of 2.79% at 100 samples and 3.03% at 200 samples.

This highlights the ability of pretrained embeddings to leverage prior linguistic knowledge when data is scarce.

Larger Sample Sizes (500–1,000):

The gap narrows as the sample size increases, with differences reducing to 1.22% at 500 samples and 0.69% at 1,000 samples.

Custom embeddings start to perform more competitively as they benefit from task-specific feature learning with increasing data.

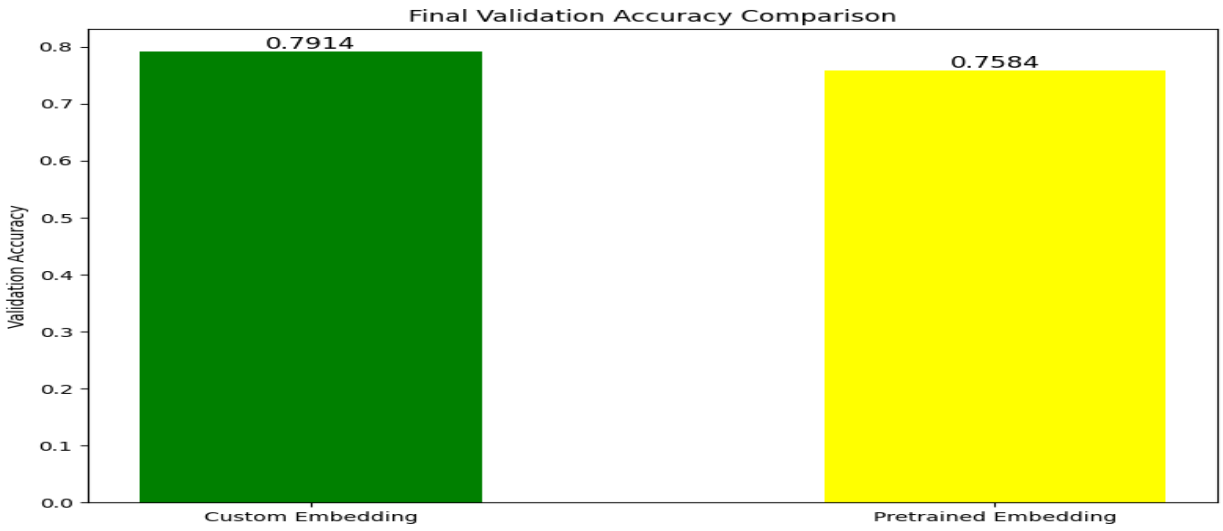
Improvement in Accuracy with Increased Sample Size:

Both models show consistent improvements in accuracy as the sample size increases.

Custom Embeddings improved from **0.7528 (100 samples)** to **0.8177 (1,000 samples)** an absolute increase of 6.49%.

Pretrained Embeddings improved from **0.7807 (100 samples)** to **0.8246 (1,000 samples)** an absolute increase of 4.39%.

The improvement trend is more pronounced for the custom embeddings, indicating that these models require more data to generalize effectively.



Plateauing Effect:

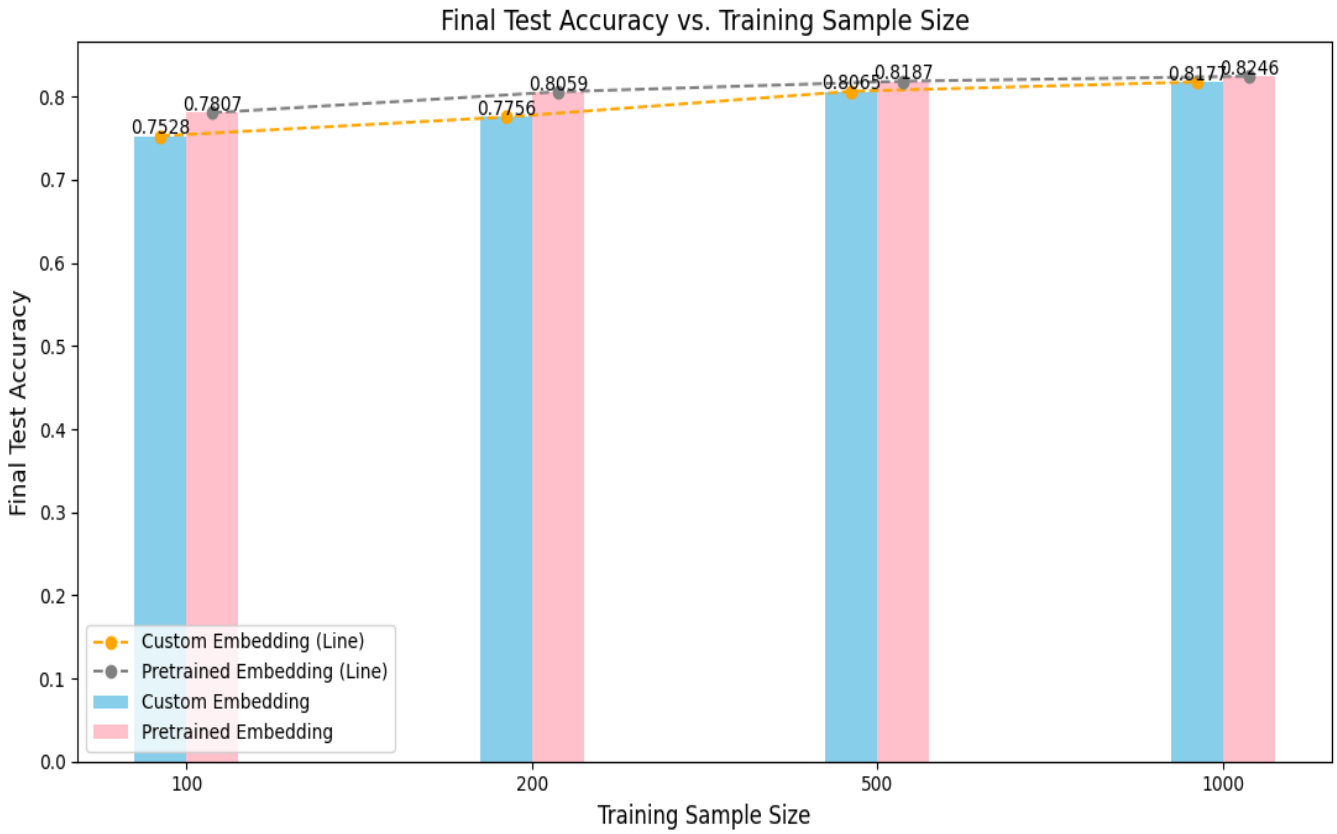
The improvements in accuracy for both approaches begin to plateau as the sample size approaches 1,000.

For instance, the increase in accuracy from 500 to 1,000 samples is smaller compared to earlier sample size increments:

Custom: **0.8065** → **0.8177** (+1.12%)

Pretrained: **0.8187** → **0.8246** (+0.59%)

This suggests diminishing returns for additional data and indicates the need for other enhancements like hyperparameter tuning or model architecture changes.



3. Key Takeaways

1. **Performance of Pretrained Embeddings**:
    - Pretrained embeddings excel in low-data scenarios due to their prior knowledge of semantic and syntactic relationships captured during pretraining on large corpora.
    - These embeddings generalize well, even with limited task-specific data, leading to better performance.
2. **Custom Embeddings' Reliance on Data**:
    - Custom embeddings require more task-specific data to learn meaningful representations, making them less effective at smaller sample sizes.
    - However, with sufficient data, they can achieve competitive performance, potentially outperforming pretrained embeddings in some scenarios.
3. **Transition Point**:
    - At **500–1,000 samples**, the performance gap between the two approaches becomes negligible. This marks a transition point where task-specific learning from custom embeddings starts catching up to the general knowledge of pretrained embeddings.
4. **Scalability**:
    - Pretrained embeddings are beneficial for projects with constraints on data availability.
    - Custom embeddings are better suited for tasks with access to larger datasets and specific domain requirements.

4. Strategic Recommendations

1. **Small Data Scenarios (<500 samples)**:
    - Use pretrained embeddings to ensure better generalization and performance.
    - Fine-tune the embeddings if possible to adapt them to the task-specific domain.
2. **Moderate to Large Data Scenarios (>1,000 samples)**:
    - Consider using custom embeddings, as they can match or surpass pretrained embeddings with sufficient data.
    - Experiment with additional model improvements such as:
        - Increasing the embedding dimensions.

        - Adding more LSTM units or additional layers.
        - Implementing attention mechanisms to capture context better.
3. **Hybrid Approaches**:
    - Investigate hybrid strategies like initializing embeddings with pretrained weights and allowing them to be trainable during the task-specific training process. This combines the benefits of both approaches.

5. Conclusion

The analysis demonstrates the clear advantage of pretrained embeddings in low-data scenarios and their robust generalization capabilities. However, with increasing data availability, custom embeddings can leverage task-specific nuances to achieve comparable or superior performance. The choice of approach should consider the dataset size, domain requirements, and computational constraints.